



Ecole Polytechnique de l'Université de Tours  
Département Informatique  
64 avenue Jean Portalis  
37200 Tours, France  
Tél. +33 (0)2 47 36 14 14  
[polytech.univ-tours.fr](http://polytech.univ-tours.fr)

## Projet Recherche & Développement 2021-2022

# Capture smart d'images sur la station TV

Projet Recherche et Développement RFAI1



Tuteur académique  
Mathieu DELALANDRE

Étudiant  
Théo REMON (DI5)

3 avril 2022



# Liste des intervenants

Nom	Email	Qualité
Théo REMON	<a href="mailto:theo.remon@univ-tours.fr">theo.remon@univ-tours.fr</a>	Étudiant DI5
Mathieu DELALANDRE	<a href="mailto:mathieu.delalandre@univ-tours.fr">mathieu.delalandre@univ-tours.fr</a>	Tuteur académique, Département Informatique



# Avertissement

Ce document a été rédigé par Théo Remon susnommé l'auteur.

L'Ecole Polytechnique de l'Université de Tours est représentée par Mathieu Delalandre susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnaît assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assume l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnaît qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.



## Pour citer ce document

Théo Remon, *Capture smart d'images sur la station TV: Projet Recherche et Développement RFAII*, Projet Recherche & Développement, Ecole Polytechnique de l'Université de Tours, Tours, France, 2021-2022.

```
@mastersthesis{
  author={Remon, Théo},
  title={Capture smart d'images sur la station TV: Projet Recherche et Développement
    RFAII},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université de Tours},
  address={Tours, France},
  year={2021-2022}
}
```

# Table des matières

Liste des intervenants	<b>a</b>
Avertissement	<b>b</b>
Pour citer ce document	<b>c</b>
Table des matières	<b>i</b>
Table des figures	<b>iv</b>
Liste des tableaux	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1 Acteurs, enjeux .....	1
2 Contexte.....	1
3 Objectifs.....	1
4 Bases méthodologiques.....	2
<b>2 Description générale</b>	<b>3</b>
1 Environnement du projet .....	3
1.1 Station TV .....	3
1.2 DELL PowerEdge T640 .....	3
2 Caractéristiques des utilisateurs.....	4
3 Fonctionnalités du système .....	4
4 Structure générale du système.....	4
<b>3 Analyse et conception</b>	<b>6</b>
1 Analyse.....	6

1.1	Détermination de la taille des empreintes et de la méthode d'interpolation.	6
1.2	Détermination du nombre de FPS minimum de capture et de l'intervalle de sélection.....	7
1.3	Choix de l'architecture de l'application .....	7
1.3.1	Architectures envisagées au début du projet.....	7
1.3.2	Architecture retenue .....	8
2	Modélisation proposée.....	9
<b>4</b>	<b>Mise en œuvre</b>	<b>10</b>
1	Réalisation de l'application .....	10
1.1	Module <code>datetime_range</code> : intervalles de dates et heures .....	10
1.2	Module <code>xmltv</code> : programmation télévisuelle et fichiers XMLTV .....	10
1.3	Module <code>rfai1</code> : partie principale de l'application .....	11
2	Choix des seuils pour les filtres sur le contraste et l'entropie .....	11
2.1	Étude de la distribution des métriques de qualité .....	11
2.2	Performance des filtres en fonction des seuils .....	13
2.3	Choix des seuils .....	13
3	Performance du filtre sur les métadonnées .....	14
4	Performance des filtres .....	14
<b>5</b>	<b>Bilan et conclusion</b>	<b>16</b>
1	Bilan du semestre 9 .....	16
2	Bilan du semestre 10.....	16
3	Bilan sur la qualité .....	16
4	Bilan auto-critique.....	16
	<b>Annexes</b>	<b>17</b>
<b>A</b>	<b>Gestion de projet</b>	<b>18</b>
1	Diagrammes de Gantt.....	18
2	Outils utilisés.....	18
<b>B</b>	<b>Spécifications fonctionnelles</b>	<b>20</b>
<b>C</b>	<b>Spécifications non fonctionnelles</b>	<b>21</b>
1	Contraintes de développement et conception .....	21
1.1	Langage de programmation.....	21
2	Contraintes de fonctionnement et d'exploitation.....	21
2.1	Ressources CPU .....	21
2.2	Volume de stockage .....	21

<b>D</b>	<b>Description des interfaces externes du logiciel</b>	<b>23</b>
1	Interfaces matériel/logiciel .....	23
2	Interfaces homme/machine.....	23
3	Interfaces logiciel/logiciel .....	23
<b>E</b>	<b>Guide d'installation et d'utilisation</b>	<b>24</b>
1	Installation .....	24
2	Utilisation .....	24
2.1	Exécution .....	24
2.2	Configuration .....	25
<b>F</b>	<b>Bibliographie</b>	<b>26</b>
<b>G</b>	<b>Acronymes</b>	<b>27</b>



# Table des figures

## 2 Description générale

- 2.1 Diagramme du système de capture intelligente d'images..... 4
- 2.2 Diagramme du programme de filtrage et de sélection d'images ..... 5

## 3 Analyse et conception

- 3.1 Graphique de l'erreur quadratique moyenne en fonction de la taille de l'image (en nombre de pixels) pour différentes méthodes d'interpolation..... 7
- 3.2 Graphique du contraste estimé en fonction du temps..... 8

## 4 Mise en œuvre

- 4.1 Graphique de comparaison entre la distribution du contraste des images crawlées (en bleu) et capturées (en rouge) ..... 12
- 4.2 Graphique de comparaison entre la distribution de l'entropie des images crawlées (en bleu) et capturées (en rouge) ..... 12
- 4.3 Graphique de comparaison entre la distribution du contraste des images capturées (en bleu) et une gaussienne ..... 13

## A Gestion de projet

- A.1 Diagramme de Gantt initial pour le semestre 9 ..... 18
- A.2 Diagramme de Gantt initial pour le semestre 10..... 18
- A.3 Diagramme de Gantt final pour le semestre 9..... 19
- A.4 Diagramme de Gantt final pour le semestre 10..... 19



# Liste des tableaux

## 3 Analyse et conception

- 1 Temps de réponse moyens pour les différentes méthodes d'interpolation ..... 7

## 4 Mise en œuvre

- 1 Moyennes et écart-type des métriques de qualité pour les images crawlées et capturées ..... 12
- 2 Pourcentage d'images passant le filtre sur le contraste en fonction du seuil ..... 13
- 3 Pourcentage d'images passant le filtre sur l'entropie en fonction du seuil ..... 13
- 4 Pourcentage d'images passant les filtres sur le contraste et l'entropie en fonction des seuils ..... 14
- 5 Couverture journalière moyenne ..... 15

# 1

## Introduction

### 1 Acteurs, enjeux

Le projet RFAI1 est un **Projet Recherche et Développement (PRD)** réalisé par Théo Remon, élève de 5<sup>ème</sup> année en spécialité informatique. Il est encadré par Mathieu Delalandre, enseignant chercheur au sein du **LIFAT**. Le sujet de ce **PRD** est proposé par l'équipe **Reconnaissance des Formes et Analyse d'Images (RFAI)** du **Laboratoire d'Informatique Fondamentale et Appliquée de Tours (LIFAT)**.

La station TV est un projet de l'équipe **RFAI** du **LIFAT** visant à mettre en œuvre de la détection de contenu et du traitement d'images et de vidéos sur les chaînes **TNT**.

### 2 Contexte

La station TV est un projet de l'équipe **RFAI** du **LIFAT** visant à mettre en œuvre de la détection de contenu et du traitement d'images et de vidéos sur les chaînes **TNT**.

Elle est à ce jour composée d'une antenne **TNT**, de baies de tuners et de machines telles que détaillées à la section **1.1**(Chapitre 2).

La station TV est et a été l'objet de plusieurs projets étudiants : cette année seulement, un **PRD**, un projet de spécialité **IA** et plusieurs projets de spécialité **ASR** ont lieu en lien avec la station. De plus, un doctorant, M. Hao Le, travaille sur une thèse en lien avec la station.

### 3 Objectifs

Le projet RFAI1 a pour sujet la réalisation d'une application de capture intelligente d'images sur la station TV. En effet, on souhaite capturer les images les plus pertinentes des programmes TV, c'est-à-dire les images de meilleure qualité.

Plus précisément, on souhaite réaliser cette capture sur la plage horaire suivante : de 6 h 00 à 2 h 00 le lendemain.

De plus, on a retenu lors de la phase de spécification le contraste comme la métrique de qualité sur laquelle on réalisera la sélection des meilleures images.

### 4 Bases méthodologiques

La méthodologie de gestion de projet retenue est le cycle en V.

On utilisera des diagrammes de Gantt pour le suivi de projet.

# 2

## Description générale

### 1 Environnement du projet

#### 1.1 Station TV

Comme mentionné précédemment, l'environnement du projet est la station TV. Celle-ci est composée de :

- une machine DELL T7610, permettant la capture simultané de 8 chaînes TNT ;
- une machine DELL PowerEdge T640, permettant le traitement simultané de 24 chaînes TNT ;
- une baie de 24 tuners TNT ;
- une baie de 8 tuners TNT pilotables grâce à des modules de télécommande infrarouge réalisées lors d'un projet Smart System par MM. Anthony LONDAIS et Clément HAUDEBOURG[1].

Les équipements sus-mentionnés sont installés dans la salle des doctorants, au bâtiment Portalis de Polytech Tours.

Une antenne situé sur le batiment Portalis, à proximité de la salle des doctorant, permet de recevoir les chaînes TNT. Cette antenne est reliée aux tuners, qui peuvent eux-même être branchés aux machines.

#### 1.2 DELL PowerEdge T640

Le programme sera exécuté sur la machine DELL PowerEdge T640. Cette machine est équipée d'un système d'exploitation Windows 10 et des composants suivant :

- 2 CPU Intel Xeon Gold 5218R, pour un total de 40 cœurs et 80 fils d'exécution ;
- 6 cartes d'acquisition AVerMedia CE314-HN, permettant chacune de traiter 4 flux vidéos, pour un total de 24 flux.

On dispose également de 2 baies de tuners :

- la première baie, équipée de 24 tuners TNT ;
- la deuxième baie, équipée de 8 tuners TNT pilotables grâce à des modules de télécommande infrarouge réalisées lors d'un projet.

## 2 Caractéristiques des utilisateurs

Les utilisateurs amenés à interagir avec le programme sont des chercheurs et étudiants des départements informatique et informatique industrielle.

Ils sont capables d'utiliser une invite de commande.

Les utilisateurs auront besoin des droits d'accès administrateur sur la machine pour pouvoir exécuter l'application.

## 3 Fonctionnalités du système

Suite à la phase de spécification et en vue de l'architecture retenue (décrite ci-dessous), le système à réaliser a été défini comme le programme de filtrage et de sélection d'images.

Ce programme sélectionnera une image par intervalle de sélection (par exemple, 30 secondes ou une minute). L'image sélectionnée sera choisie par un filtrage et une sélection réalisés grâce aux métadonnées (date et heures de capture) et aux métriques de qualité (contraste et entropie) des images.

## 4 Structure générale du système

L'architecture retenue pour le système de capture intelligente d'images est composée des programmes suivants :

- un programme capturant les images, avec les caractéristiques suivantes :
  - capture à faible FPS,
  - capture sur 24 chaînes TNT,
  - images enregistrées au format JPEG ;
- un programme filtrant et sélectionnant les images pertinentes, avec les composants suivants :
  - filtre sur les métadonnées (date et heure de capture) des images, pour ne garder que les images correspondant à un programme TV (autrement dit, éliminer les images correspondant aux publicités et jingles),
  - filtre avec un seuil minimum pour l'entropie des images,
  - sélecteur sur le contraste (une image – celle ayant le contraste le plus élevé – est choisie par intervalle de sélection).

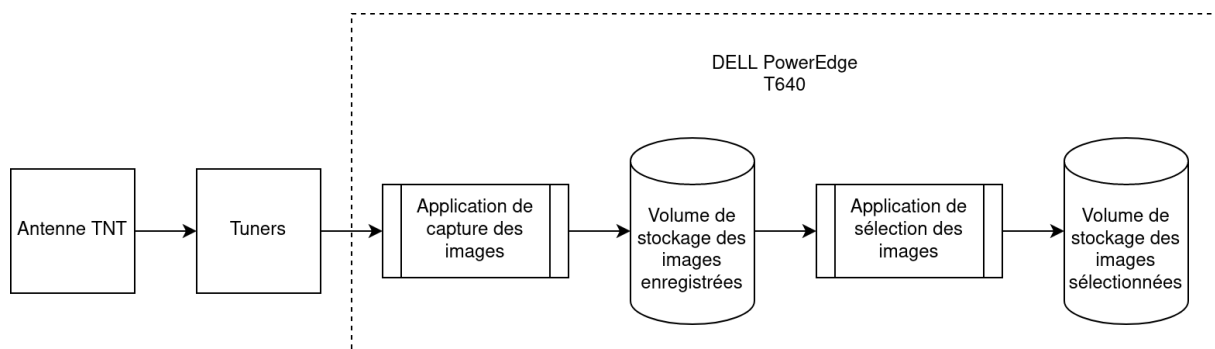


Figure 2.1 – Diagramme du système de capture intelligente d'images

Le premier programme est réalisé par d'autres étudiants dans le cadre de leur projet ASR.

Le second programme sera l'objet de la partie réalisation de ce PRD.

La réalisation du premier programme et le déploiement des programmes n'entrent pas dans le cadre de ce projet.



**Figure 2.2** – *Diagramme du programme de filtrage et de sélection d'images*

# 3

## Analyse et conception

### 1 Analyse

#### 1.1 Détermination de la taille des empreintes et de la méthode d'interpolation

Dès le début du projet, il était clair que calculer les métriques de qualité sur les images elles-mêmes demanderait trop de puissance de calcul.

En effet, les architectures envisagées demandaient que le programme calcule les métriques pour 8 à 24 chaînes TNT, selon l'architecture. Les architectures envisagées demandant un calcul en temps réel, il a donc été décidé de calculer les métriques de qualité sur les empreintes des images (c'est-à-dire, les images redimensionnées vers une plus petite taille, pour laquelle le calcul des métriques de qualité est moins coûteux).

Dès lors, il fallut déterminer la meilleure taille pour les empreintes. Pour cela, deux critères ont été considérés : premièrement, la taille de l'empreinte, car celle-ci détermine le coût de calcul, et l'erreur entre la métrique de qualité actuelle (autrement dit, calculée sur l'image) et estimée (calculée sur l'empreinte de l'image), car on souhaite que le calcul sur les empreintes soit fiable. Pour mesurer l'erreur, la métrique d'erreur quadratique moyenne a été choisie.

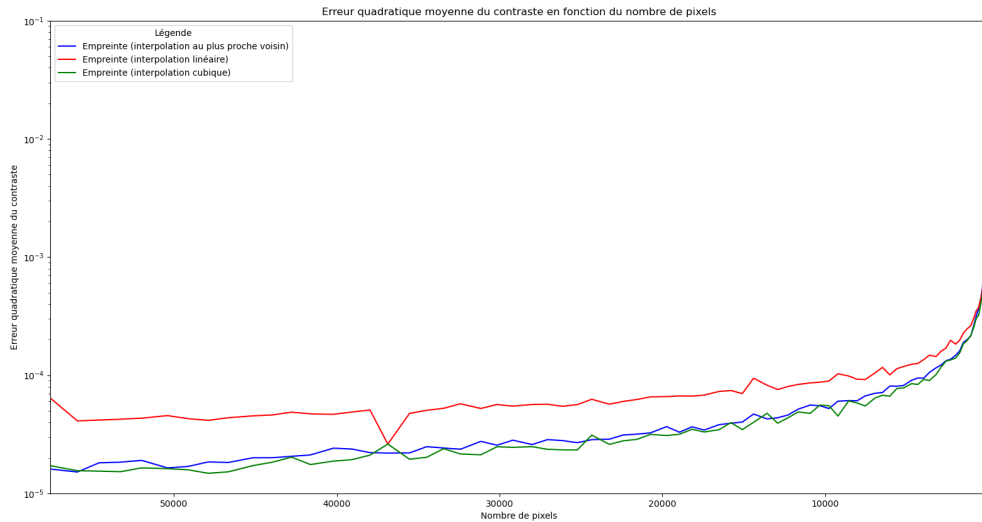
Un programme a été réalisé pour calculer l'erreur quadratique moyenne du contraste (la métrique de qualité retenue pour la sélection d'images) pour différentes tailles d'empreintes d'images et pour différentes méthodes d'interpolation. Ce programme a été réalisé en Python et utilise la bibliothèque OpenCV.

Ainsi, le graphique 3.1 a pu être établi. Il représente l'erreur quadratique moyenne en fonction de la taille des empreintes pour les tailles d'empreintes allant jusqu'à  $320 \times 180 = 57600$  pixels et pour les méthodes d'interpolation au plus proche voisin, linéaire et cubique.

Au vu de ces résultats, la taille des empreintes de images a été définie comme  $96 \times 54$  pixels (nombre de pixels : 5184; erreur quadratique moyenne :  $3.823 \cdot 10^{-5}$ ).

Un benchmark du temps de réponse moyen pour le calcul des empreintes avec les différentes méthodes d'interpolation a été réalisé (fils d'exécution : 1; itérations : 100000; taille des empreintes :  $96 \times 54$ ). Le tableau 1 correspond aux résultats du benchmark.

Au vu des résultats du benchmark, il a été déterminé que l'interpolation au plus proche voisin sera utilisée pour le calcul des empreintes des images.



**Figure 3.1** – Graphique de l'erreur quadratique moyenne en fonction de la taille de l'image (en nombre de pixels) pour différentes méthodes d'interpolation

**Table 1** – Temps de réponse moyens pour les différentes méthodes d'interpolation

Méthode	Temps de réponse moyen
Au plus proche voisin	5.40 $\mu$ s
Linéaire	11.62 $\mu$ s
Cubique	27.53 $\mu$ s

## 1.2 Détermination du nombre de FPS minimum de capture et de l'intervalle de sélection

La deuxième partie de l'analyse a consisté en l'étude de l'évolution de la métrique de qualité choisie (le contraste) dans le temps pour déterminer le nombre de FPS minimum de capture et l'intervalle de sélection.

Premièrement, une capture a été réalisée avec les caractéristiques suivantes : une chaîne TNT, durée de 90 minutes, 60 FPS. Cette capture a permis d'acquérir un ensemble de 324020 images pour lesquelles on a calculé le contraste. On a ainsi obtenu le graphique 3.2.

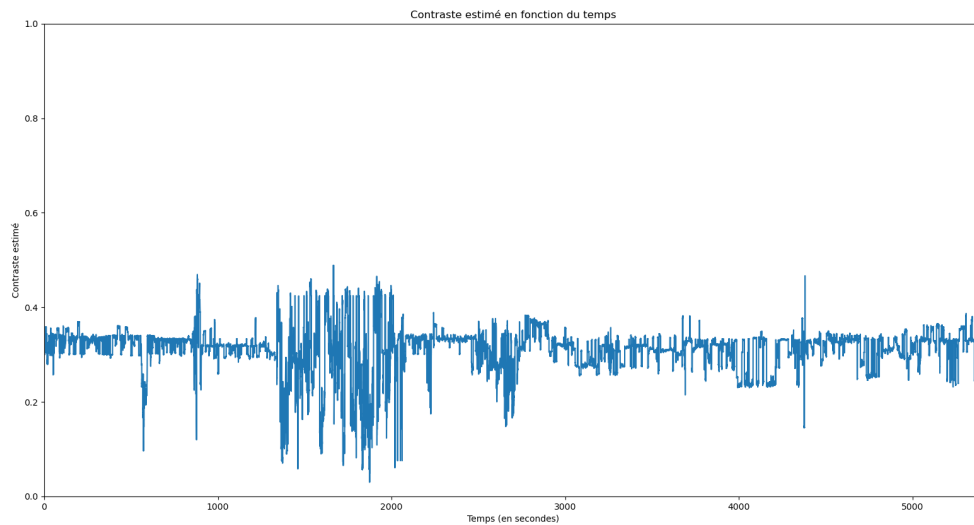
Grâce à ce graphique, il a été observé que les pics de contraste durent généralement au moins une seconde.

## 1.3 Choix de l'architecture de l'application

### 1.3.1 Architectures envisagées au début du projet

Au début du projet, trois architectures proposées par M. Delalandre étaient envisagées. Ces architectures consistent en un seul programme qui s'interface avec les cartes de captures, récupère les flux, calcule les métriques et enregistre les images sélectionnées.

Les architectures envisagées étaient les suivantes :



**Figure 3.2** – *Graphique du contraste estimé en fonction du temps*

**1. Architecture A :**

- capture sur 24 chaînes statiques (24 flux Full HD) ;
- avec la baie de 24 tuners ;
- calcul des métriques en temps réel.

**2. Architecture B :**

- capture sur 12 chaînes statiques (12 flux Full HD et 12 flux d’empreintes) ;
- avec la baie de 24 tuners ;
- calcul des métriques en temps réel.

**3. Architecture C :**

- capture sur 4 chaînes dynamiques (4 flux Full HD et 4 flux d’empreintes) ;
- avec la baie de 8 tuners pilotables (qui permettent de changer les chaînes) ;
- calcul des métriques en temps réel.

L’architecture A permettrait de couvrir la majorité de l’offre **TNT** (24 chaînes). L’architecture B en revanche offrirait une couverture nettement inférieure (12 chaînes).

L’architecture C permettrait de couvrir l’ensemble de l’offre **TNT**. Néanmoins, la couverture de chaque chaîne serait partielle (pour 24 chaînes, chaque chaîne aurait un taux de couverture de  $\frac{1}{6}$ , sans compter le temps mis par les tuners pour changer de chaîne).

L’architecture C permettrait d’allouer plus de ressources pour le calcul des métriques de qualité de chaque chaîne – et donc d’utiliser de métriques plus coûteuse – que l’architecture B, supérieure elle-même à l’architecture A sur ce critère.

### 1.3.2 Architecture retenue

Des tests de temps de réponse réalisés par M. Le ont fait apparaître que la puissance de calcul de la station TV ne permet pas le calcul des empreintes et des métriques en temps réel.

Ainsi, il a fallu adopter une nouvelle architecture où le calcul des métriques n’est pas réalisé en temps réel :

**1. Architecture D :**

- **1<sup>er</sup> programme :**

- capture d’images sur 24 chaînes à faible FPS (à cause de la limite en écriture sur le disque) ;
- avec la baie de 24 tuners ;
- pas de calcul des métriques en temps réel.
- **2<sup>ème</sup> programme :**
  - filtrage des images sur leurs métadonnées (date et heure de capture) avant de les charger depuis le disque ;
  - chargement des images depuis le disque ;
  - calcul des métriques en différé par rapport à la capture ;
  - filtrage et sélection sur les métriques calculée.

Le premier programme a pour seul rôle la capture des images. Le filtrage et la sélection des images est réalisé par le deuxième programme après que la capture soit effectuée.

## 2 Modélisation proposée

La programme de sélection et de filtrage fonctionnera de la manière suivante :

1. Filtrage des images par leur date et heure de capture :
  - à l’aide d’un fichier XMLTV détaillant les programmes des chaînes TNT, on filtre les images de façon à ne garder que les images appartenant à un programme ;
  - ce premier filtre permet :
    - d’éliminer les images auxquelles nous ne sommes pas intéressés (publicités, jingles, etc),
    - de réduire les accès disques en lecture (les images ne passant pas ce filtre d’étant pas chargées),
    - de réduire la charge de calcul (le programme ne calculera pas les métriques de qualités sur ces images).
2. Filtrage des images avec un seuil minimum sur l’entropie :
  - l’entropie estimée des images est calculée ;
  - les images dont l’entropie est supérieure ou égale au seuil passent le filtre, les autres ne passent pas ;
  - ce deuxième filtre permet l’élimination des images à faible entropie, que l’on considère pour les besoin de ce programme comme de mauvaise qualité.
3. Sélection des images sur le contraste :
  - on sélectionne une image par intervalle de sélection ;
  - le contraste estimé des images est calculé ;

Les termes contraste estimé et entropie estimée d’une image désignent respectivement le contraste et l’entropie calculés sur une empreinte de l’image obtenue.

Tel que déterminé durant la phase de spécification, les empreintes des images seront calculées par interpolation au plus proche voisin.

La phase de spécification a également permis de déterminer un bon compris entre la taille des empreintes (qui détermine le coût de calcul des métriques de qualité sur celles-ci) et le taux d’erreur (plus précisément, l’erreur quadratique moyenne). La taille d’empreinte déterminée est de  $96 \times 54$  pixels.

# 4

## Mise en œuvre

La partie mise en œuvre de ce **PRD** a principalement consisté en :

- la réalisation de l'application ;
- la définition des paramètres des filtres.

### 1 Réalisation de l'application

L'application a été réalisé en Python (version 3.10) avec les bibliothèques **NumPy** et **OpenCV**.

Elle prend la forme de trois modules Python :

- `datetime_range` ;
- `xmltv` ;
- `rfai1`.

#### 1.1 Module `datetime_range` : intervalles de dates et heures

L'application est amenée à travailler avec des intervalles de dates et heures, au niveau du filtre sur les métadonnées, pour vérifier que l'heure de diffusion d'une image soit dans une plage horaire correspondant à un programme.

J'ai donc réalisé le module `datetime_range` pour fournir un modèle simple permettant de manipuler des intervalles de dates et heures.

#### 1.2 Module `xmltv` : programmation télévisuelle et fichiers XMLTV

Pour filtrer les images sur leur heure de diffusion, et donc sur le fait qu'elles appartiennent ou non à un programme, il faut récupérer la programmation télévisuelle.

Cela se fait par le biais de fichiers XMLTV, qui décrivent la programmation télévisuelle dans un fichier XML.

C'est là qu'intervient le module `xmltv`. En effet, il offre des modèles pour les chaînes, les programmes (émissions) et les programmations, et permet d'importer et d'exporter des programmations télévisuelles depuis et vers des fichiers XMLTV.

Les fonctionnalités d'import et d'export reposent sur le module `xml` de la bibliothèque standard de Python.

### 1.3 Module `rfai1` : partie principale de l'application

Le module `rfai1` correspond à la partie principale de l'application et représente la majorité du code produit.

Ce module est lui-même organisé de sous modules, qui correspondent aux différentes fonctionnalités de l'application. Ceux-ci sont :

- `rfai1.application` : la partie qui met en œuvre les différentes composantes de l'application ;
- `rfai1.filters` : les différents filtres de l'application (sur les métriques de contraste et d'entropie, sur les métadonnées) ;
- `rfai1.image` : le modèle pour les images, pouvant optionnellement embarquer des métadonnées ;
- `rfai1.metadata` : le modèle pour les métadonnées ;
- `rfai1.selectors` : le sélecteur sur le contraste ;
- `rfai1.settings` : les paramètres de l'application, avec un importeur ;
- `rfai.__main__` : le lanceur de l'application quand celle-ci est appelée en ligne de commande (`python -m rfai1`).

C'est dans ce module que les filtres sur les métadonnées, sur le contraste et sur l'entropie ainsi que le sélecteur ont été réalisés.

## 2 Choix des seuils pour les filtres sur le contraste et l'entropie

Pour choisir les seuils pour filtres sur les métriques de qualité (contraste et entropie), je me suis intéressé à deux jeux d'images :

- les images crawlées, qui correspondent à des images que l'on souhaite garder ;
- les images capturées, qui correspondent aux images capturées sur une chaîne (France 2) sur une période de six jours à 2 FPS, pour un total d'environ 1.1 millions d'images.

### 2.1 Étude de la distribution des métriques de qualité

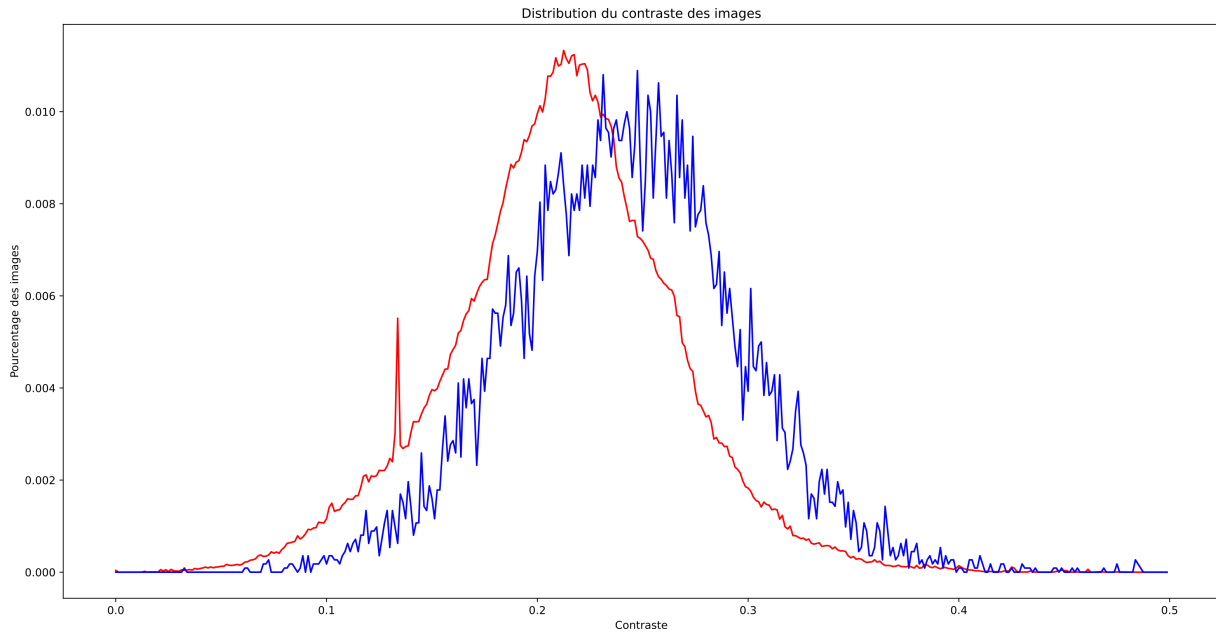
Je me suis tout d'abord intéressé à la distribution du contraste et de l'entropie pour ces deux jeux d'images.

Sur les graphiques 4.1 et 4.2, on peut observer que les courbes de distributions des métriques de qualité des images crawlées et des images capturées se chevauchent.

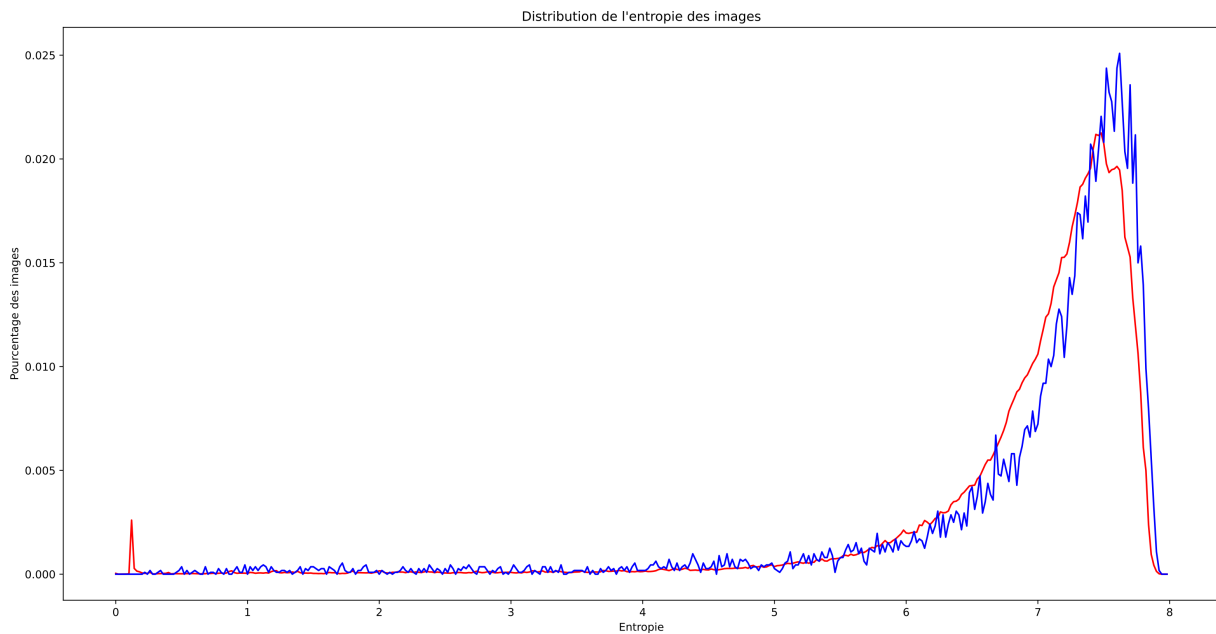
De plus, les courbes de distribution du contraste semblent être proches de courbes gaussiennes. Les courbes de distribution de l'entropie semblent quant à elles être proches de courbes demi-gaussiennes.

J'ai donc calculé les moyennes et écart-types pour les deux métriques de qualité (voir tableau 1).

J'ai ainsi pu comparer la courbe de distribution du contraste des images capturées à la courbe gaussienne correspondant à sa moyenne et à son écart-type (voir graphique 4.3).



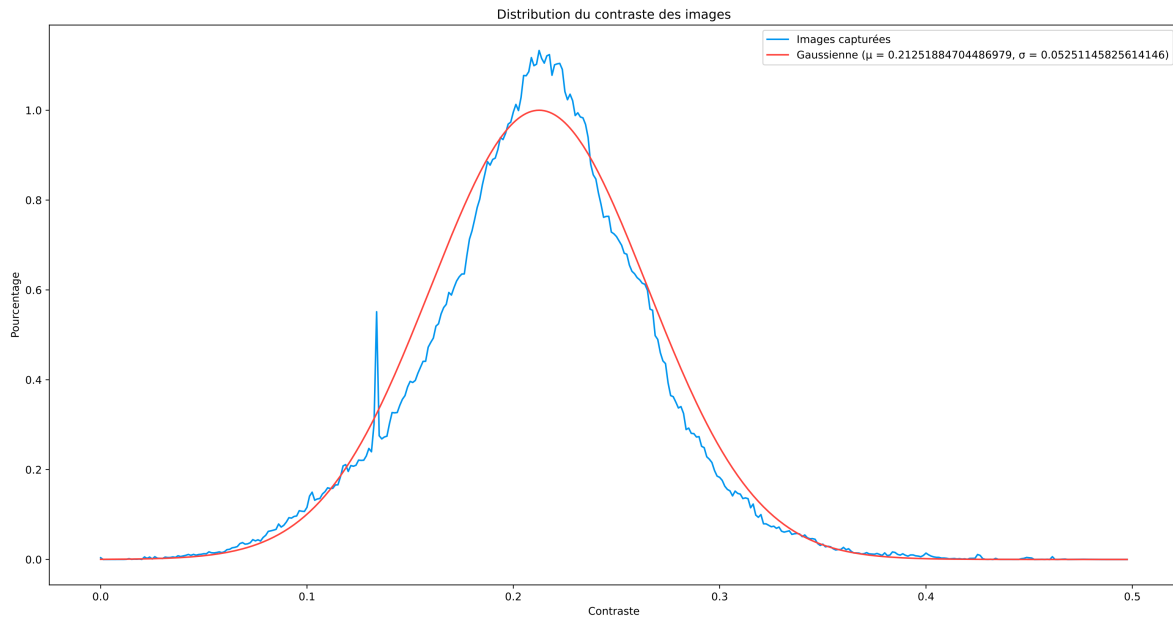
**Figure 4.1** – Graphique de comparaison entre la distribution du contraste des images crawlées (en bleu) et capturées (en rouge)



**Figure 4.2** – Graphique de comparaison entre la distribution de l'entropie des images crawlées (en bleu) et capturées (en rouge)

**Table 1** – Moyennes et écart-type des métriques de qualité pour les images crawlées et capturées

Jeu d'images	Contraste		Entropie	
	Moyenne ( $\mu_c$ )	Écart-type ( $\sigma_c$ )	Moyenne ( $\mu_e$ )	Écart-type ( $\sigma_e$ )
Image crawlées	0.242207	0.053489	7.003225	1.111502
Image capturées	0.212519	0.052511	7.006561	0.924988



**Figure 4.3** – Graphique de comparaison entre la distribution du contraste des images capturées (en bleu) et une gaussienne

## 2.2 Performance des filtres en fonction des seuils

La prochaine étape a été de s'intéresser au volume d'image passant les filtres selon la valeur des seuils.

J'ai utilisé les  $\mu_c$ ,  $\mu_e$ ,  $\sigma_c$  et  $\sigma_e$  des images crawlées, qui correspondent à des images types que l'on souhaite capturer.

Les résultats de ces tests de performances sont présentés dans les tableaux 2, 3 et 4.

**Table 2** – Pourcentage d'images passant le filtre sur le contraste en fonction du seuil

Valeur du seuil		Pourcentage d'images passant le filtre
$\mu_c - \sigma_c$	0.188717	70.06%
$\mu_c - 2 \cdot \sigma_c$	0.135228	92.24%
$\mu_c - 3 \cdot \sigma_c$	0.081739	99.11%

**Table 3** – Pourcentage d'images passant le filtre sur l'entropie en fonction du seuil

Valeur du seuil		Pourcentage d'images passant le filtre
$\mu_e - \sigma_e$	5.891723	93.17%
$\mu_e - 2 \cdot \sigma_e$	4.780221	97.27%
$\mu_e - 3 \cdot \sigma_e$	3.668719	98.36%

## 2.3 Choix des seuils

L'objectif du filtrage étant de diminuer le nombre d'images, on choisira de fixer les seuils à  $\mu - \sigma$ . Ainsi, les valeurs des seuils choisies sont :

**Table 4** – Pourcentage d’images passant les filtres sur le contraste et l’entropie en fonction des seuils

Valeur des seuils		Pourcentage d’images passant le filtre
Contraste	Entropie	
$\mu_c - \sigma_c$	$\mu_e - \sigma_e$	68.22%
$\mu_c - 2 \cdot \sigma_c$	$\mu_e - 2 \cdot \sigma_e$	91.01%
$\mu_c - 3 \cdot \sigma_c$	$\mu_e - 3 \cdot \sigma_e$	97.74%

— seuil sur le contraste : 0.18871738992757967

— seuil sur l’entropie : 5.891722906947287

Les filtres sur le contraste et l’entropie laisseront donc passer environ 68% des images.

Il est à noter que filtrer sur le contraste et l’entropie ne permet pas de réaliser un filtrage précis sur la qualité des images. Des faux positifs passeront donc ces filtres.

### 3 Performance du filtre sur les métadonnées

Le filtre sur les métadonnées permet éliminer les images n’appartenant pas aux programmes, comme par exemple les images des publicités.

On souhaite donc savoir quelle pourcentage d’une journée est couverte par des programmes afin d’estimer les performances de ce filtre.

Les programmes ne commencent pas exactement à l’heure à laquelle ils sont programmés. M. Le a établi un intervalle de latence de  $[-590, 820]$  (en secondes).

Afin de prendre en compte la latence, on ne considère une image comme faisant partie d’un programme que si elle appartient à l’intervalle  $[t_d + 820, t_f - 590]$ , où  $t_d$  et  $t_f$  sont respectivement les heures de début et de fin du programme (les programmes de durée inférieure à 1410 secondes ne sont donc pas considérés).

On exclut également la plage horaire 2 h 00 – 6 h 00.

On mesure ensuite le temps moyen de couverture journalière par chaîne (voir tableau 5).

On obtient un taux de couverture moyen de 42%. Il est à noter que le taux de couverture varie significativement selon la chaîne.

Le filtre sur les métadonnées devrait donc laisser passer 42% des images en moyenne.

### 4 Performance des filtres

La performance des filtres cumulés (métadonnées, contraste et entropie) est d’environ  $68.22 \times 42.02 = 28.67\%$ .

**Table 5** – *Couverture journalière moyenne*

Chaîne	Durée	Pourcentage
TF1	10 h 58 min 33 s	45.73%
France 2	8 h 58 min 49 s	37.42%
France 3	6 h 51 min 8 s	28.55%
France 4	2 h 55 min 49 s	12.21%
France 5	8 h 28 min 50 s	35.34%
France Info	9 h 17 min 6 s	38.69%
M6	10 h 27 min 25 s	43.57%
Arte	10 h 2 min 25 s	41.83%
C8	11 h 12 min 5 s	46.67%
W9	12 h 58 min 37 s	54.07%
TMC	13 h 29 min 25 s	56.21%
TFX	9 h 58 min 52 s	41.59%
NRJ 12	10 h 5 min 19 s	42.04%
La Chaîne parlementaire	10 h 39 min 49 s	44.43%
BFMTV	14 h 55 min 52 s	62.21%
CNEWS	10 h 2 min 2 s	41.81%
Total	10 h 5 min 8 s	42.02%

# 5

## Bilan et conclusion

### 1 Bilan du semestre 9

Lors du semestre 9, j'ai pu me familiariser avec le système et établir les spécifications de l'application qui sera l'objet de la partie réalisation.

Pour cela, j'ai réalisé une étude visant à déterminer la taille des empreintes des images ainsi que la méthode d'interpolation qui sera utilisée pour les calculer. J'ai également réalisé une étude permettant de définir le nombre minimum d'images à capturer par seconde par le programme de capture.

J'ai également réalisé le cahier de spécification.

Il reste à réaliser l'application de capture et de sélection.

### 2 Bilan du semestre 10

Lors du semestre 10, j'ai procédé à la réalisation de l'application et à l'évaluation des performances des filtres.

L'application que j'ai réalisé correspond aux spécifications établies lors du semestre 9.

### 3 Bilan sur la qualité

L'application que j'ai réalisé est bien commentée et versionnée en utilisant Git.

J'ai nommé mes variables et fonctions de manière à bien décrire leur contenu.

La partie test pourrait être améliorée : ma procédure de test n'est pas documentée ou automatisée, ce qui pourrait entraîner des difficultés pour quelqu'un qui souhaiterait modifier mon application.

### 4 Bilan auto-critique

Les parties spécification et réalisation de mon projet se sont bien passées. Néanmoins, je pourrais m'améliorer sur la partie planification.

## Annexes

# A

# Gestion de projet

## 1 Diagrammes de Gantt

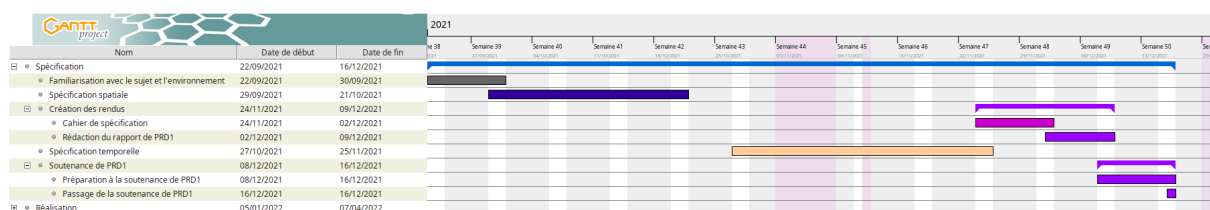


Figure A.1 – Diagramme de Gantt initial pour le semestre 9

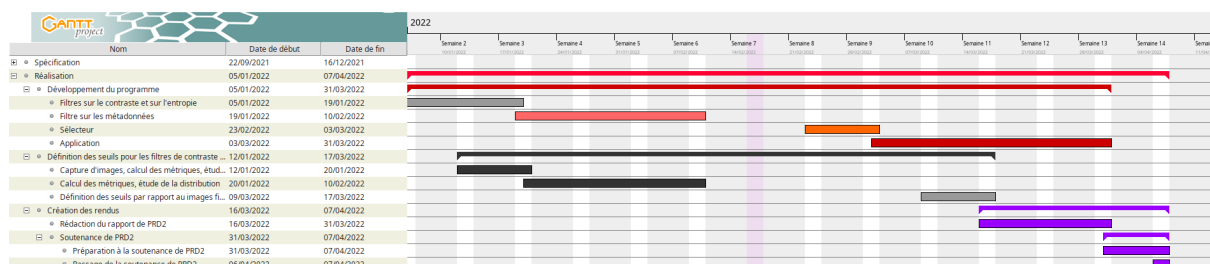


Figure A.2 – Diagramme de Gantt initial pour le semestre 10

## 2 Outils utilisés

Le logiciel de gestion de version Git a été utilisé pour le versionnage des fichiers sources produits lors du PRD (modules Python de l'application, fichiers sources L<sup>A</sup>T<sub>E</sub>X des documents produits, etc).

Le logiciel GanttProject a permis de réaliser les différents diagrammes de Gantt.

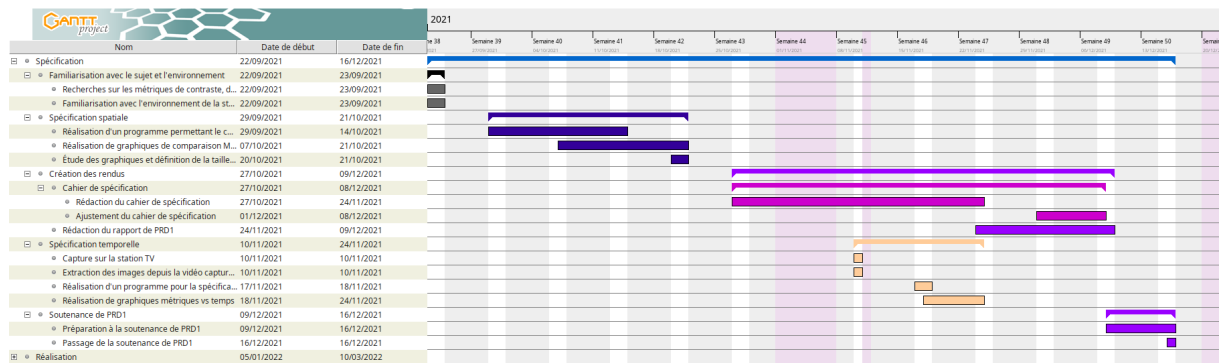


Figure A.3 – Diagramme de Gantt final pour le semestre 9

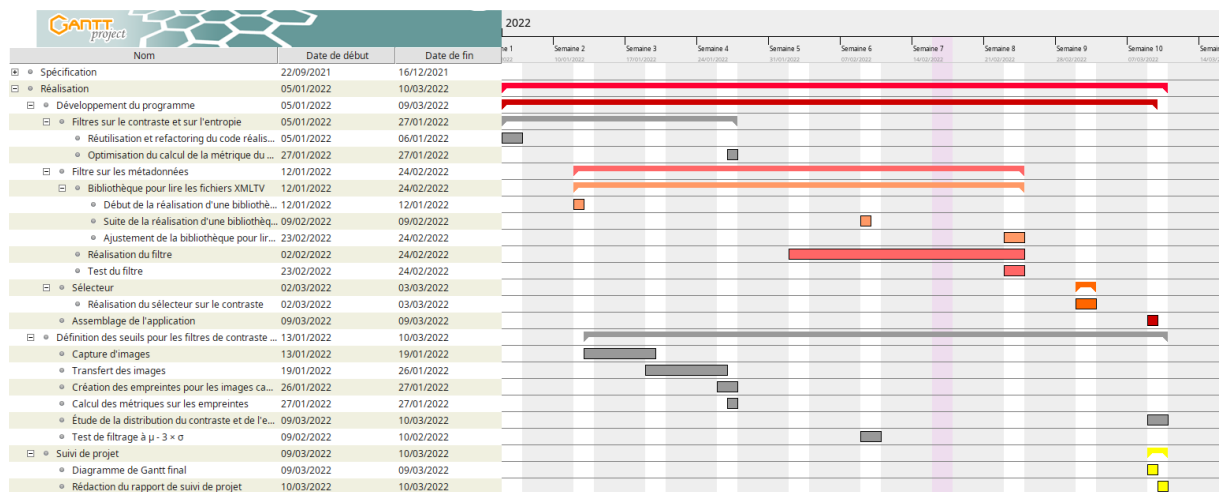


Figure A.4 – Diagramme de Gantt final pour le semestre 10

# B

## Spécifications fonctionnelles

Le programme de filtrage et de sélection, qui est l'objet de la réalisation du **PRD**, doit répondre à trois fonctions :

### 1. Filtrage d'images sur les métadonnées

- **Description** : Filtre les images grâce à leur date et heure de capture (métadonnées), en vérifiant si elle appartiennent au contenu d'un programme et non à une publicité, à un jingle, etc.
- **Entrées** :
  - un ensemble d'images  $X$  ;
  - l'ensemble des dates et heures de capture  $T$  tel que  $t_i \in T$  soit la date et l'heure de capture de l'image  $x_i \in X$ .
  - un fichier XMLTV
- **Sorties** :
  - un ensemble d'images  $Y$  composé des images de  $X$  qui appartiennent au contenu d'un programme.

### 2. Filtrage d'images sur l'entropie

- **Description** : Filtre les images avec un seuil sur leur entropie estimée.
- **Entrées** :
  - un ensemble d'images  $X$  ;
  - un seuil  $s_H$ , valeur minimale d'entropie pour qu'une image passe le filtre.
- **Sorties** :
  - un ensemble d'images  $Y$  composé des images  $x_i$  de  $X$  dont l'entropie  $H(x_i)$  est supérieure au seuil  $s_H$  (c'est-à-dire,  $H(x_i) \geq s_H$ ).

### 3. Sélection d'images sur le contraste

- **Description** : Sélectionne les images les plus pertinentes. La sélection est réalisée sur le contraste estimé des images. Une image est sélectionnée par intervalle de capture.
- **Entrées** :
  - un ensemble d'images  $X$  ;
  - l'ensemble des dates et heures de capture  $T$  tel que  $t_i \in T$  soit la date et l'heure de capture de l'image  $x_i \in X$  ;
  - un intervalle de capture sur lequel on sélectionne une image.
- **Sorties** :
  - un ensemble d'images  $Y$  composé des images de  $X$  sélectionnées.

# C

# Spécifications non fonctionnelles

## 1 Contraintes de développement et conception

### 1.1 Langage de programmation

Les opérations sur les images (calculs de contraste et d'entropie, interpolation) étant réalisé via OpenCV, il est possible d'utiliser les langages de programmation pour lesquels une interface existe, c'est-à-dire C++, Java, MATLAB et Python.

La librairie OpenCV étant écrite en C++, utiliser Python n'impactera pas significativement les performances.

## 2 Contraintes de fonctionnement et d'exploitation

### 2.1 Ressources CPU

La machine DELL PowerEdge T640 dispose d'une puissance de calcul limité : 40 cœurs à fréquence maximale de 4 GHz et 80 fils d'exécution.

Les opérations nécessaires à la sélection et à l'enregistrement des images en temps réel et sur de multiples chaînes risquent de dépasser la puissance de calcul disponible sur la machine.

Pour remédier à ce problème, on dispose de plusieurs solutions :

- calculer les métriques de qualité (contraste et entropie) des images sur leur empreintes : ceci peut réduire massivement le coût de calcul des métriques ;
- réaliser un filtrage préalable sur les métadonnées (date et heure de diffusion) pour éliminer les images non pertinentes (jingles, publicités).

### 2.2 Volume de stockage

Un disque de 12 TB a été alloué à ce projet. Celui-ci ne devrait poser de problème au niveau de l'espace de stockage disponible que pour les très longues campagnes de capture (de l'ordre d'une ou plusieurs années).

Le facteur limitant pour le programme de sélection et de filtrage sera la vitesse de lecture sur le disque.

# D

## Description des interfaces externes du logiciel

### 1 Interfaces matériel/logiciel

Le programme de sélection et de filtrage n'interagit pas avec le matériel.

### 2 Interfaces homme/machine

Le programme de filtrage et de sélection sera un script Python que l'on pourra lancer en ligne de commande.

Les différents paramètres seront passés au programme sous la forme d'un fichier de configuration. Le chemin du fichier de configuration sera passé en argument au programme.

### 3 Interfaces logiciel/logiciel

Le programme de filtrage et de sélection n'interagira pas avec des logiciels externes.

Cependant, il utilisera la bibliothèque OpenCV pour interagir avec les images.

# E

# Guide d'installation et d'utilisation

## 1 Installation

L'application requiert Python en version 3.10 ou ultérieure. Il y a trois bibliothèques Python à installer :

- la bibliothèque pour les intervalles de dates et heures ;
- la bibliothèque pour les fichiers XMLTV ;
- la bibliothèque pour l'application.

Ces bibliothèques sont respectivement fournies dans les dossiers suivants :

- `python-datetime-range`
- `python-xmltv`
- `python-rfai1`

Il faut installer la bibliothèque de l'application en dernier. Pour installer une bibliothèque, il faut ouvrir un terminal (ou équivalent) dans le dossier de la bibliothèque et exécuter la commande suivante :

```
python -m pip install .
```

## 2 Utilisation

### 2.1 Exécution

L'application peut être exécutée avec la commande suivante :

```
python -m rfai1 [ARGUMENTS]
```

Elle prend un argument optionnel et quatre arguments obligatoires :

- le chemin du fichier XMLTV décrivant les programmes ;
- l'identifiant de la chaîne dans le fichier XMLTV ;
- le dossier contenant les images à traiter. Celui-ci ne doit contenir que les images correspondant à la chaîne ;
- le dossier où les images sélectionnées seront copiées. Celui-ci doit être vide ;

- optionnel : le chemin du fichier de configuration, valant par défaut "`settings.xml`", au format :
  - "`-s PATH`"
  - ou "`-settings PATH`"

Pour plus d'information, entrer la commande :

```
python -m rfai1 -help
```

## 2.2 Configuration

Le fichier de configuration par défaut, au format XML, est fourni dans le dossier de la bibliothèque pour l'application, sous le nom "`settings.xml`". Il est possible d'y modifier :

- les dimensions des empreintes ;
- les seuils minimaux et maximaux pour les filtres de contraste et d'entropie ;
- les décalages au début et à la fin des programmes ;
- l'intervalle de sélection ;
- le nombre de threads.



## Bibliographie

- [1] Clément HAUDEBOURG et Anthony LONDAIS. *Projet Station TV RAPPORT PROJET SMART SYSTEM*. 2021.



## Acronymes

**ASR** Architecture Systèmes et Réseaux

**FPS** Frames Per Second

**IA** Intelligence Artificielle

**JPEG** Joint Photographic Experts Group

**LIFAT** Laboratoire d'Informatique Fondamentale et Appliquée de Tours

**PRD** Projet Recherche et Développement

**RFAI** Reconnaissance des Formes et Analyse d'Images

**TNT** Télévision Numérique Terrestre

## Objectifs

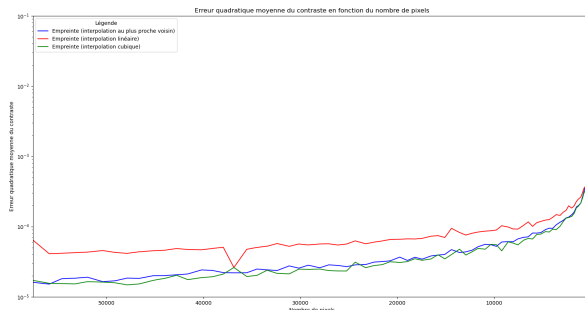
Le **PRD** RFAI1 consiste en la mise en œuvre d'une application de capture intelligente d'images sur les chaînes **TNT**

## Mise en œuvre

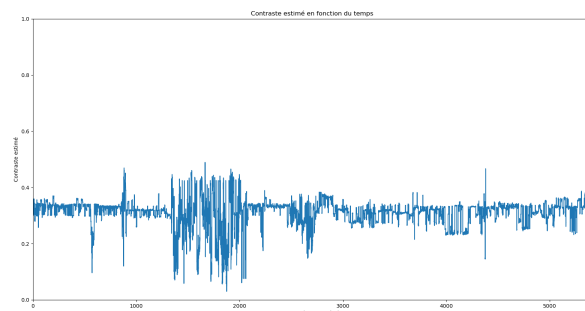
Une application permettant le filtrage d'images sur les métadonnées et des métriques de qualité et la sélection des meilleures images a été réalisée.

## Résultats attendus

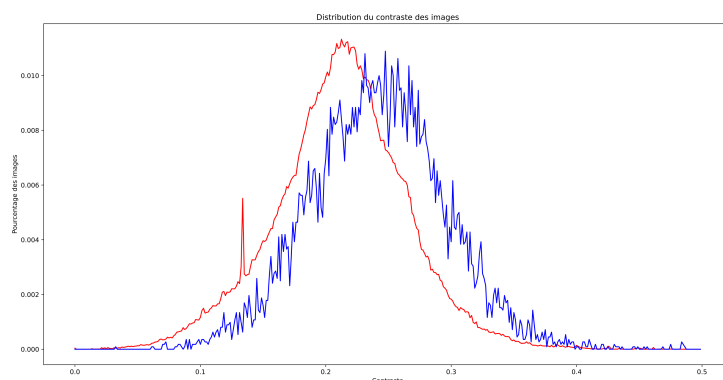
Un programme permettant de sélectionner les meilleures images depuis un ensemble d'images capturées.



Graphique de l'erreur quadratique moyenne en fonction de la taille de l'image (en nombre de pixels) pour différentes méthodes d'interpolation



Graphique de l'évolution du contraste en fonction du temps



Graphique de comparaison entre la distribution du contraste des images crawlées (en bleu) et capturées (en rouge)

## Objectifs

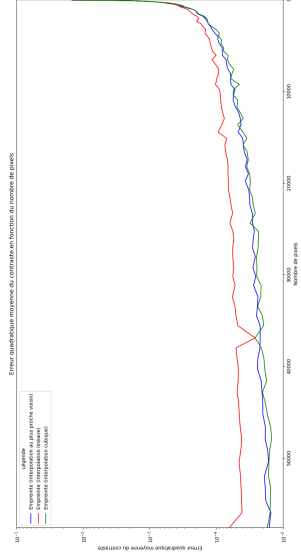
Le **PRD** RFAI1 consiste en la mise en œuvre d'une application de capture intelligente d'images sur les chaînes **TNT**

## Mise en œuvre

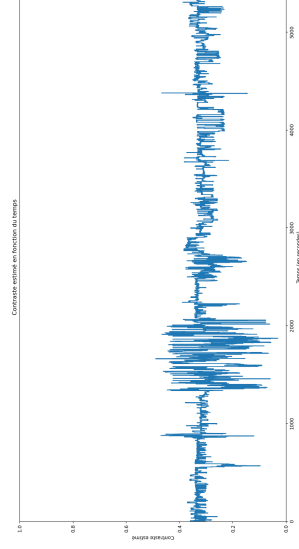
Une application permettant le filtrage d'images sur les métadonnées et des métriques de qualité et la sélection des meilleures images a été réalisée.

## Résultats attendus

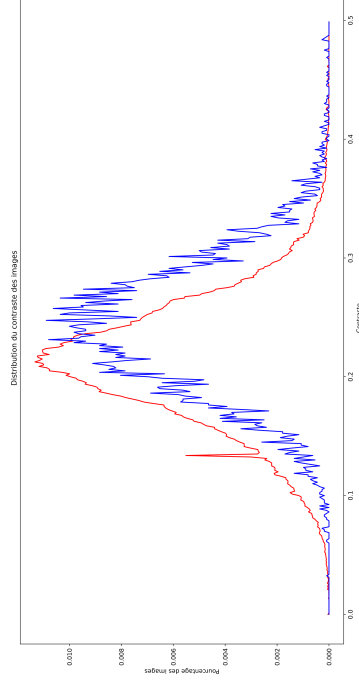
Un programme permettant de sélectionner les meilleures images depuis un ensemble d'images capturées.



Graphique de l'erreur quadratique moyenne en fonction de la taille de l'image (en nombre de pixels) pour différentes méthodes d'interpolation



Graphique de l'évolution du contraste en fonction du temps



Graphique de comparaison entre la distribution du contraste des images capturées (en bleu) et sélectionnées (en rouge)

# Capture smart d'images sur la station TV

## Projet Recherche et Développement RFAI1

### Résumé

Ce **Projet Recherche et Développement** est proposé par le groupe **RFAI** du **Laboratoire d'Informatique Fondamentale et Appliquée de Tours**. Il consiste en la mise en place d'un système de capture intelligente d'images depuis les chaînes **TNT** sur la station TV.

### Mots-clés

capture intelligente d'images, station tv, tnt

### Abstract

This Research and Development Project was proposed by the **RFAI** group of the **Laboratoire d'Informatique Fondamentale et Appliquée de Tours**.

### Keywords

image smart capture, tnt, tv workstation